IDS#124733 (990.0525) Attorney Docket No. Atkinson 3-1-28

APPLICATION FOR UNITED STATES LETTERS PATENT

FOR

RESTORATION TIME IN MESH NETWORKS

Inventors: Gary W. Atkinson
Michael L. Craner
Ramesh Nagarajan

Prepared by: Mendelsohn & Associates, P.C.
1515 Market Street, Suite 715
Philadelphia, Pennsylvania 19102
(215) 557-6656
Customer No. 22186

* * * * *

Certification Under 37 CFR 1.10

"Express Mail" Mailing Label No. EV140153268US Date of Deposit March 31, 2004.

I hereby certify that this document is being deposited with the United States Postal Service's "Express Mail Post Office To Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

Mary E. Caniz
(Name of person mailing)

Mary E. Caniz
(Signature of person mailing)

RESTORATION TIME IN MESH NETWORKS

CROSS-REFERENCE TO RELATED APPLICATIONS

This application claims the benefit of the filing date of U.S. provisional application no. 60/459,163, filed on 03/31/2003, incorporated herein by reference in its entirety. The subject matter of this application is related to U.S. patent application no. 10/673,383 filed on 09/26/2003 as attorney docket no. Doshi 57-6-22-18-34, incorporated herein by reference in its entirety (herein "Doshi '03") and U.S. patent application no. 10/673,056 filed on 09/26/2003 as attorney docket no. Alfakih 1-1-1-6-2, also incorporated herein by reference in its entirety (herein "Alfakih '03")

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates to optical networks and, more specifically, to cost reduction and restoration time improvement in mesh optical networks.

Description of the Related Art

Reliability and cost are two parameters that drive the design of modern day networks. To support high reliability, it is typical to provide path redundancy for services. To control costs, it is common to attempt to maximize the utilization of available resources and generate redundant paths in consideration of multiple-cost criteria.

Generally, reliability is supported by providing both a primary path and a restoration path for each service in the network. In the event of a failure along the primary path for a service, the service is switched over to the associated restoration path. For optical mesh networks, one challenge is to support restoration times that are comparable to those provided by SONET/SDH networks with self-healing rings (e.g., 10-100ms restoration times). To help reduce network restoration time in optical mesh networks, a number of approaches have been considered, including improving restoration signaling and associated algorithms, and improving the switching speed of cross-connection infrastructure switching elements.

SUMMARY OF THE INVENTION

One factor that determines a lower bound on restoration time is the maximum number of cross-connections to be performed at a single network element in the event of a failure. Assuming everything else is constant, the larger the number of cross-connections, the longer the restoration time. Thus, path planning can be performed which minimizes restoration time by carefully selecting primary and restoration paths for demands within a network such that the worst-case number of cross-connections is minimized across all possible single-event failures in the network.

On the other hand, the choice of primary and restoration paths for demands in a network can also affect the cost of a network where the magnitude of the effect on cost can be different depending on the cost metric that is considered (e.g., administrative weight, bandwidth, distance cost, and/or the degree to which restoration bandwidth between multiple disjoint primary paths can be shared).

Problems in the prior art are addressed in accordance with principles of the present invention by a method and apparatus for restoration-path planning that minimizes network cost while meeting an upper bound on restoration time associated with single-event failures in a mesh network.

In certain embodiments, network-cost minimization is a function of maximizing the utilization of network resources and can include the maximization of sharing of network resources (e.g., restoration bandwidth), while restoration time is bounded by application of an optimization that reduces the worst-case number of cross-connections that must be performed in a network in the event of a single element (e.g., node or link) failure.

One optimization involves two phases. The first phase of the optimization involves finding two node-disjoint paths for each service demand within a network such that the maximum link bandwidth in the network is minimized and the link bandwidths within the network are substantially leveled. The second phase involves identifying the primary and restoration paths for each service demand within the network such that the worst-case number of cross-connections at any node within the network is minimized across all possible single-event failures.

BRIEF DESCRIPTION OF THE DRAWINGS

Other aspects, features, and advantages of the present invention will become more fully apparent from the following detailed description, the appended claims, and the accompanying drawings in which:

FIG. 1 illustrates a method for minimizing cost within a network while meeting restoration-time requirements via the minimization of cross-connections within the network according to one embodiment of the present invention.

FIG. 2 illustrates another method for minimizing cost within a network while meeting restoration-time requirements via the minimization of cross-connections within the network according to another embodiment of the present invention.

DETAILED DESCRIPTION

Reference herein to "one embodiment" or "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiment can be included in at least one embodiment of the invention. The appearances of the phrase "in one embodiment" in various places in the specification are not necessarily all referring to the same embodiment, nor are separate or alternative embodiments mutually exclusive of other embodiments.

Cost and Restoration Time

Network designers typically are faced with more than one constraint in the process of network design. Although it may be important to a service provider to find a network path plan that meets the restoration-time goals of a client, it is also important, from a competitive perspective, for the service provider to do so with minimum cost. To this end, one embodiment of the present invention combines restoration-time-minimization techniques such as those discussed in Alfakih '03 with resource-sharing-maximization techniques such as those discussed in Doshi '03. As can be appreciated by one skilled in the art, there are a number of different ways to implement a solution to this problem. In the following, two exemplary procedures are discussed.

Cost Relaxation with Fixed Restoration-Time Bound

One embodiment of the present invention is illustrated by the procedure of **FIG. 1**. As shown in steps **102** and **104**, the network topology and traffic demands, respectively, for the network are input. In step **106**, the variable MinXC# is set to the graph-theoretical minimum number of cross-connections (XCs) for the network topology and traffic demands that were input in steps **102** and **104**. Also, in step **106**, the variable *k* is initialized to CostDelta, a value (e.g., one) that represents the cost relaxation step size that will be used by the procedure. In step

108, set *A* is set equal to the set of the primary/restoration path plans for the network that minimize cost (e.g., maximizes sharing per Doshi '03).

Next, in step **110**, the variable MinCost is set equal to the maximum cost of any of the path plans in set *A*. The "cost" can be a bandwidth cost, administrative cost, distance cost, or a combination of one or more of those or other cost metrics that are important to the network planner, other than restoration time, which is considered separately in this implementation.

In step **112**, set *B* is set equal to the set of primary/restoration path plans for the network whose worst-case number of XCs required at any one node after considering all possible single-event failures is less than or equal to XC#max, where XC#max is a bound on the maximum number of cross-connections that can be tolerated in the network according to some quality-of-service agreement between the service provider and the client. So, for example, if the service agreement is a bound of 300ms, and the time for a single cross-connection is 10ms, XC#max would be 30. XC#max, in actual implementations, may be a function of a number of different variables including traffic, message bundle sizes, signaling bandwidth, signaling processor speed, and message buffers sizes, as would be understood to one skilled in the art. XC#max is always greater than or equal to MinXC# and is sometimes initialized to be some predefined offset from MinXC#, for example XC#max = MinXC# + XC#delta.

Next, in the test of step **114**, set *C* is assigned to be the intersection of sets *A* and *B* and this intersection is compared with the null set {0}. If *C* is not equal to the null set, then this means that each of the path plans in *C* has a maximum number of cross-connections that is within the service agreements of the network operator and also has minimum cost. In this case, the method terminates at step **116** with a set *C* of one or more path plans that meet the restoration time goals of the service provider at minimal cost.

If the intersection of the two sets in step **114** is null, then, in step **118**, set *A* is set equal to the set of all primary/restoration path plans whose cost is less than (MinCost + *k*). In other words, the cost constraint is relaxed by one unit of CostDelta. In step **120**, *k* is incremented by CostDelta, and, in step **122**, the value of *k* is checked against the variable CostDeltaLimit, which had been initialized to a value (e.g., 20) that would prevent the procedure from selecting a network plan whose cost was beyond allowable bounds. If the value of *k* is too large, then the procedure completes with an error in step **124**. If not, then processing returns to step **114** as described earlier.

In one or more embodiments, step **108** of **FIG. 1** can involve use of algorithms that use multiple-cost criteria for primary/restoration path selection (see, e.g., Doshi '03), and step **112** can involve use of a multiple-phase optimization that yields path plans that exhibit a minimum number of cross-connections required at any given node in the event of a failure (see, e.g., Alfakih '03). As described in Alfakih '03, the first phase involves finding two node-disjoint paths for each service demand within a network such that the maximum link bandwidth in the network is minimized and the link bandwidths within the network are leveled. The second phase involves further constraining a commercial solver with cost goals and identifying the primary and restoration paths for each service demand within the network such that the worst-case number of cross-connections at any node within the network is minimized across all possible single-event failures within the cost constraints.

Dual Phase Solver

A two-phase embodiment of the present invention is illustrated by the procedure of **FIG. 2**. In step **202**, the network topology and traffic demands, respectively, for the network are input. In step **204**, a parameter WC#XC (worst-cast number of cross-connects) is set to a value between (1) the quantity MinXC# (minimum cross-connect number), a graph theoretical minimum number of cross-connections given the network topology and traffic demands that were input in step **202**, and (2) the quantity WC#XC_Limit, a limit on the largest value of the WC#XC parameter that would be considered acceptable based on, e.g., a service-agreement requirement on network restoration time.

In step **206** (phase I), a commercial solver is run to find two node-disjoint paths for each service demand within the network such that the maximum link bandwidth in the network is minimized and the link bandwidths within the network are substantially leveled. (Two paths are referred to as node-disjoint if they have no intermediate (i.e., transit) nodes in common.) Note that one parameter used by the solver is the current value of WC#XC. Its role in the solver is to essentially limit the solution space from which the solver can select the path pairs.

In step **208**, a test is performed to see if the solver found a feasible solution in step **206** (given the current value of WC#XC). If not, then, in step **210**, the value of WC#XC is increased by a small increment (i.e., the constraint is relaxed somewhat), and the new value of WC#XC is then tested in step **212** to determine if it exceeds the limit WC#XC_Limit. If it does exceed the limit, then the procedure terminates in step **214**. Otherwise, the procedure returns to step **206**.

For a suitably large value of WC#XC_Limit, in step **206**, a feasible solution will eventually be found and the test of step **208** will then pass. In this case, step **216** (phase II) is performed. Here, a solver is used for path planning. The result of the solver is that one path of each node-disjoint path pair that was found in step **206** is assigned to be a primary path for the pair's associated demand, and the other path is assigned to be the restoration path for the demand. This path planning is performed to find a path plan from within the solution space, which path plan has the characteristic that the worst-case number of cross-connections at any node within the network is minimized across all possible single-event failures.

In step **218**, a test is performed to see if the worst-case number of cross-connections of the resulting path plan is less than or equal to the current value of WX#XC. If the test of step **218** passes, then, in step **220**, a cost for the path plan is calculated, and the plan and its cost are saved. Next, in step **222**, a constraint is formulated and added to the primary/restoration path assignment that prevents the identical path plan from being obtained the next time step **216** is performed. The path planning is then reattempted in step **216** with the additional constraint.

Eventually, the test in step **218** will fail when no more path plans can be found in step **216** that satisfy the constraint that the worst-case number of cross-connections is less than or equal to WC#XC. In this case, as shown in step **224**, all the solution-limiting constraints previously added in step **222** are eliminated. In step **226**, constraints or conditions are added to the load-balancing problem of step **206** to prevent the previous path-pair solution from being generated. The modified load-balancing problem is then re-solved in step **206** at the current value of WC#XC.

This process is repeated for increasing values of WC#XC until eventually the value exceeds WC#XC_Limit at step **212**, in which case the procedure terminates in step **214**. At this point, assuming WC#XC_Limit was chosen sufficiently large, a number of primary/restoration path plans have been stored along with their costs. These plans all have an acceptable number of worst-case cross-connections (i.e., no worse than WC#XC_Limit). The user can search these path plans for the minimum-cost plan, or can perform a tradeoff between cost and restoration time within this limited set of plans.

Note that, alternatively or additionally, step **216** can include constraints that drive the solver toward minimum-cost path plans. These constraints can be derived from multiple-cost criteria including consideration of sharing of restoration bandwidth between disjoint path failures as discussed in Doshi '03.

Path Costs

Generally, there are a number of different metric that can be applied in calculating paths for demands within a network. Various algorithms including shortest path, minimum bandwidth, and fewest hops have been proposed in the art. As networks become more complex, these cost criteria become more complex as well. For example, the cost of path administration and reconfiguration costs of various types (measured in, for example, restoration time) can also be considered along with bandwidth costs and administrative costs. Further, multiple cost criteria, weighted relative to their importance to the path-planning algorithm, can be considered concurrently. As discussed in Doshi '03, link state information (e.g., bandwidth allocation) and sharing information can be used to compute cost-efficient connection routes within a network. For path computation, link state can be reduced to an equivalent link cost and then route-computation algorithms can be considered that minimize the total path cost – where the path cost is considered to be the sum of the link costs. Depending on the possibility of sharing bandwidth on the restoration path, link costs will be different when restoration bandwidth can be shared compared to the when it cannot. (Note that when network bandwidth can be shared, sharing information can be used to compute more cost-efficient paths through the network. This can be achieved by incorporating sharing information in the link-cost metric). The two cases are described below.

No-Sharing

Link-cost calculation can be based on the administrative weight (AW) of the link, the link capacity (LC), and link's available capacity (AC). Under light and medium link utilization (LU), where LU is less than a specified utilization threshold (UT) (i.e., $LU \leq UT$), link cost w is set equal to the AW, i.e., $w = AW$. Hence, under light load conditions, the link-calculation algorithm will assign links to services according to the preferences (i.e., administrative weights) assigned by the network operator. When the link load is high ($LU>UT$), however, the link weights are preferably calculated with the objective of load balancing and revenue maximization. The generic formula for the link cost w in this high link utilization region is based on the inverse of available capacity:

$$w = \frac{AW \cdot MWC}{AC^f} \quad (1)$$

where MWC is the maximum weight coefficient (i.e., an additional scaling factor introduced to provide continuity between the two regions of light and heavy loads) and f is an exponentiation factor (nominally set to 0.4) used to modify the available capacity variable AC. Motivation, a detailed model, and additional numerical support for this approach of weight calculation based on the inverse of available capacity are described in Dziong, Z., "ATM Network Resource Management," McGraw-Hill, 1997, (herein "Dziong '97") incorporated herein by reference in its entirety.

Sharing

When sharing information is available, it can be used to compute more cost-efficient (more optimal) primary and restoration paths. For example, an algorithm can be designed to compute for each possible primary path the lowest cost restoration path by utilizing the sharing information. Then from the set of all primary and restoration paths, the pair that requires least amount of additional capacity can be chosen. This path computation algorithm using the sharing information can give considerably better paths then an algorithm using no sharing information.

Sharing information can be used in finding the least cost restoration path for a given primary path. It requires adjustment of the link cost (lowering of it) based on the amount of sharing that is possible on a link along the restoration path of a particular primary path. This can be achieved by computing the sharing degree of each link in the network given the primary path. (Note that only links that are disjoint to the primary path need to be considered).

The sharing degree is defined as the maximum number of additional (unit bandwidth) primary services (along the proposed primary path) that can be added to the link without increasing the restoration bandwidth requirement of the link. In a sense, this metric provides a network planner with an idea of the restoration headroom on a link with respect to the proposed primary path. The higher the sharing degree, intuitively, the better the choice of the primary path for the new service, since a larger sharing degree for a primary path would allow future demands to be added along that path without the need to reserve additional restoration bandwidth.

Sharing degree can be calculated from an aggregate node-link vector V_{nla} representation of sharing information and a primary path node-link vector V_{pnl} representation according to the following relationship:

SD = the maximum value m for which $\max\{ m \cdot V_{pnl} + V_{nla} \} = \text{RB}$,

where RB is the current reservation bandwidth on the link under consideration.

A less accurate measure of sharing degree can be calculated using a compact representations of the aggregate node-link vector. Note that, here, less accurate means that the sharing degree only provides a conservative indication of which links may be better, but it does not provide the exact bandwidth available for sharing for a particular primary path.

Sharing degree can be calculated from the node-aggregate vector V_{na} representation of sharing information and the primary path node vector V_{pn} representation according to the following relationship:

SD = the maximum value m for which $\max\{ m \cdot V_{pn} + V_{na} \} = \text{RB}$.

Sharing degree can also be calculated using the binary representation of node-link or node vector. In case availability of sharing information in terms of the binary node-link vector V_{nlb} for a link, the sharing degree can be computed by first deriving a binary primary path node-link vector V_{pnlb} from the primary path node-link vector V_{pnl} (in the similar fashion as V_{nlb} can be derived from V_{nla}), then taking an OR of the V_{pnlb} (binary primary path node-link vector) and V_{nlb} (binary node link vector representation of the sharing information on the link), and then taking the bit AND of the resulting vector. This will result in a sharing degree of one if the sharing is possible and zero otherwise. Note that the sharing degree obtained in this manner using the binary node-link vector does not give the exact amount of sharing that is possible on the link for the primary path. It only indicates that if the sharing is possible or not.

Similarly, a more crude sharing degree can be derived by using the binary node vector information.

In one implementation, the link cost is calculated according to the following equation, which considers the sharing degree (SD) calculated as discussed earlier:

$$w = \frac{AW}{(1+SD)}.$$

When bandwidth sharing is possible for a link, it would appear that there is no immediate bandwidth-related cost for new restoration path reservation using that link. However, when applying the Markov decision theory framework described in Dziong '97, there is a cost. This follows from the fact that the cost should be considered during the whole connection-holding time, not just at the instant of a new connection arrival. The Markov process makes it possible to consider the probabilistic cost of using the link, since, even if sharing is possible at the moment of connection arrival, in the future, with some probability, the other shared connections can be terminated and the new connection will be the sole occupant of the reserved bandwidth on that link, and hence incur a cost for reserving additional restoration bandwidth in the network. While exact calculation of such a cost seems to be very difficult, if possible, one can use this argument to employ a non-zero link cost even when sharing is possible. In one implementation, this fact can be accommodated by calculating link cost according to the following equation:

$$w = \frac{AW}{(1+b \cdot SD)} \qquad (2)$$

where b is a specified coefficient.

While the embodiments of this invention have been discussed with respect to implementations that pre-compute pathways, they may equally well be applied to implementations where some or all of the computation or optimization of alternative routes is computed after the failure is detected and in some cases may involve fault isolation in the alternative route determination.

While this invention has been described with respect to specific costs typically minimized in path planning algorithms inclusive of costs that are a function of sharability of network resources, other costs and multiple cost criteria can be considered, as would be understood to one skilled in the art.

While this invention has been described with respect to restoration associated with situations involving single-point failures, the concepts, and in particular, the link-state description, can be extended to multiple-point failure situations, as would be understood to one skilled in the art.

While this invention has been described with reference to illustrative embodiments, this description should not be construed in a limiting sense. Various modifications of the described embodiments, as well as other embodiments of the invention, which are apparent to persons skilled in the art to which the invention pertains are deemed to lie within the principle and scope of the invention as expressed in the following claims.

Although the steps in the following method claims, if any, are recited in a particular sequence with corresponding labeling, unless the claim recitations otherwise imply a particular sequence for implementing some or all of those steps, those steps are not necessarily intended to be limited to being implemented in that particular sequence.